

# TruParking: Smart Parking and the Internet of Things

Jacob Bertish  
Computer Science Department  
Truman State University  
Kirksville, United States  
jdb4276@truman.edu

William Krause  
Agriculture Science Department  
Truman State University  
Kirksville, United States  
wvk3845@gmail.com

Austin Jarrett  
Computer Science Department  
Truman State University  
Kirksville, United States  
asj4535@truman.edu

Chetan Jaiswal  
Computer Science Department  
Truman State University  
Kirksville, United States  
cjaiswal@truman.edu

**Abstract**— The number of licensed drivers and motor-vehicles has been increasing over the last several decades. This increase in the number of drivers and vehicles implies that there is more traffic on the roads which creates a greater need for more parking. However, the number of parking spaces is not increasing at the rate that traffic is increasing. This is creating a somewhat overlooked issue in the parking sector. People are now on average spending more time and resources driving around searching for a parking space. The time spent searching for parking is causing people to waste time on an action that is unproductive, meaningless, and often frustrating. The wasted time and resources should instead be put towards benefiting the people and their societies. In this paper we propose a solution for parking problems using the Internet of Things (IoT) and data analytics, we call this solution TruParking. TruParking offers a means by which users can reduce the time and resources they waste searching parking lots (or parking garages) for available parking spaces.

**Keywords**—parking, data analytics, Internet of Things, cloud, pervasive computing, ubiquitous computing

## I. INTRODUCTION

According to the Federal Highway Administration, there were approximately 112 million licensed drivers in 1970 and by 2009 that number had increased to 210 million, a compounded annual increase of 1.6 percent [1]. Recent data, from 2016, suggests that the total number of registered motor-vehicles has risen to 265 million [2] [3]. In today's world, there is an ever-increasing number of registered drivers and vehicles, because of our increasingly mobile society. Individuals use their vehicles to transport to and from work, school, social events, and essentially any activity outside of their home. While it is great that vehicles allow us to travel great distances, it comes with a variety of issues. Some of these issues include accidents, traffic jams, and fuel usage. A frequently overlooked issue is parking.

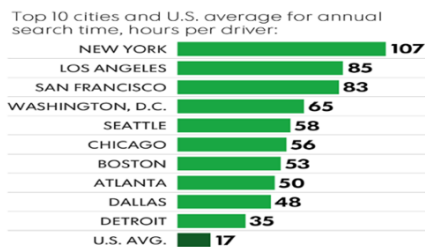


Figure 1. Data on time spent searching for parking spaces over the course of a year [4]

Figure 1 presents data included in a USA Today statement that searching for parking is more painful than ever for U.S. drivers; motorists spend an average of 17 hours a year searching for spots on streets, in lots, or in garages [4]. On an individual level, 17 hours wasted over the course of a year may not seem like an issue; however, on a societal level, this adds up to a very large amount of wasted time. Additionally, INRIX estimates that this time spent parking is costing Americans at least 73 billion dollars a year [5]. This number includes time spent parking, fuel used, and emissions. The data for New York, a city notorious for traffic-related issues, states that this issue costs the individual driver \$2,200 annually and costs the city \$4.3 billion annually. This issue isn't limited to large cities like New York; cities all over the country experience parking issues year-round. Events like Black Friday, concerts, sporting events, and nearly all other public events contribute to the parking dilemma. Most day-to-day activities require some amount of driving, and any amount of driving usually results in needing some form of parking. All of this goes to show that parking, and the associated problems, are reasonably inevitable in today's world.

Our work proposes a device that can contribute to solving the previously mentioned issues. TruParking monitors incoming and outgoing traffic through parking entrances and exits. TruParking's algorithm determines, with a level of certainty, whether a passing object is a vehicle, and if the object is determined to be a vehicle, also determines the direction of travel of the vehicle when passing the infrared (IR) sensors. This determination is then processed to provide real-time data on a given parking lot's vacant spots.

TruParking provides a simple, straightforward web interface and mobile application to present its data. Our mobile application and web interface will allow users to provide a location and then return parking availability for the nearby area.

This information will be readily available to any user connected to the Internet. Having access to this information will minimize the time spent searching for a parking spot by presenting the user with data showing which lots they will likely find a spot upon their arrival. The analytics, combined with the real-time availability aspect of TruParking, gives users the information necessary to decide whether to search a specific parking lot.

The economic and cost-effective nature of our work gives TruParking a stark advantage over most related

works. TruParking cuts costs by limiting the number of components required to provide parking information. Additionally, TruParking is a stand-alone device that does not require constant monitoring, cutting costs even further. The economic nature of our device, however, does not compromise its functionality or quality.

The remainder of the paper is as follows: Section II reviews related literature, section III outlines our contributions, section IV discusses our evaluation of our tool, section V details future-plans for our work, section VI concludes the paper.

## II. LITERATURE REVIEW

In this section, we discuss previous works that offer solutions to issues pertaining to parking spot availability.

Indect is a product that uses camera-based sensors, each monitoring up to 6 spaces, to provide information about the availability, or lack thereof in a parking garage [6]. This product offers a solution to that of monitoring the available parking spaces in each parking lot, however, the distinction between this product and TruParking is that Indect would require a separate sensor for every 6 parking spots, while TruParking only requires one device at the entrances and/or exits of a given lot. This establishes TruParking as an effective, more affordable alternative.

The work in [7] has information about an apparatus that serves a similar purpose to that of the present device in this paper. The claim in the patent that “...a search [for a parking space] is time-consuming, harmful to the ecology, and often frustrating,” supports our claim that time spent searching for a parking space is wasteful and can, and should, be mitigated. The device examined in the patent is an apparatus that uses a system of optical devices and a three-dimensional mapping of a given parking lot to return information about the location of available spaces. The advantage that TruParking holds over that of the device examined is that our device requires little to no calibration to set up an optical device and the related three-dimensional mapping. Our device would have a relatively simple installation only at each point of entry or exit for the given lot. An increase in the number of spots in the given parking lot would have no meaningful impact on our device; the only thing that would have an impact on the setup for a parking lot would be the number of entry or exit points.

The work presented in [8] uses an Arduino Uno and RFID (radio frequency identification) scanners. The purpose of this work was to develop a “*Car Park Network (CPN)*” to combat the same issues addressed in our work. Each vehicle is to be equipped with a unique RFID chip on the front license plate. The chip will be scanned to authenticate the user information, and an entrance or exit will be registered for the given lot. The availability of spots in a parking lot or parking garage is displayed on a mobile application for users. This cloud-based device is marketed as an economical option, however, with RFID, creating a large network can become expensive. The advantage TruParking has over the work in [8] is that each vehicle does not need a special tag, only one device per entrance/exit. The economical design of our product is extremely efficient and affordable.

The Internet of Things (IoT) based smart parking device shown in [9] was developed for users to be able to check the availability of parking in real-time. Along with that users are able to reserve spots ahead of time. The downfall to this is that the cost of the device is increased because sensors need to be placed in each spot and at the entrance of the parking facility. With our TruParking model, we combat this issue by only using one device per entrance or exit. Exponentially decreasing the cost of installation and implementation. The economical design of TruParking delivers real-time information as well as analytical data to plan travel ahead of time. Therefore at a fraction, of the cost the TruParking device will still provide adequate information for all users.

## III. OUR CONTRIBUTIONS

In this section, we discuss the approach we took in developing TruParking. We lay out the components of our device and explain how we have used them to contribute to the work.

### A. Our Approach

The TruParking device was developed to optimally run for but is not limited to, parking lots and parking garages.

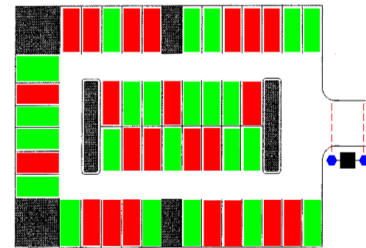


Figure 2. TruParking system demonstration

Figure 2 shows that TruParking uses a pair of simulated break-beam sensors to determine an entry or exit to the parking lot. After this is determined the device then sends the data to our cloud storage along with a timestamp. This gives us the ability to determine the availability of parking in any parking lot.

### B. Powering Our Device

To power the TruParking device the Raspberry Pi must receive 5 volts of power to boot up and run tasks. To supply the Pi with this power there are multiple options. The ability to hardwire a micro-usb from a wall outlet is possible but for our scope, this was not feasible. With many parking lots being outside we developed a solar power design to fuel our device and use natural energy to recharge our battery.

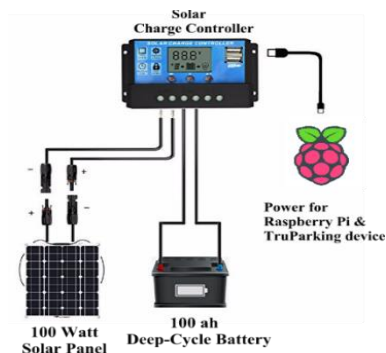


Figure 3. Solar power system for TruParking device

Figure 3 displays the connections needed to run the system properly. Our 100-amp hour deep-cycle battery powers the charge controller with 12 volts of power. The charge controller then takes this power and regulates it to keep our Pi safe from receiving too much power. Once power is drawn from the battery, the 100-Watt solar panel is used to charge the battery. The system is constantly working to make sure the battery never runs critically low. According to [10], at 400% CPU load the Raspberry Pi B 3 only consumes 730-milliamp-hour, this is equivalent to 0.73-amp-hour. Since the Pi will not be consistently running at 400% CPU load, the battery at the minimum could run for 5 days without a solar charge. Making this the optimal way of charging the device in an outdoor environment, no matter the conditions.

### C. Mechanism

Our device contains three physical parts which are the infrared (IR) rangefinders, Arduino Uno and the Raspberry Pi.



a. Front view of Sharp IR rangefinder



b. Back view of Sharp IR rangefinder

Figure 4. Sharp IR rangefinder

The first component of the device is the Sharp IR rangefinders which are shown in figures 4a and 4b. This model (GP2Y0A710K0F) of sensors are specifically configured to detect the range between 100 - 500 centimeters which is the average width of an entrance in a parking garage or lot [11]. The sensor has two components, a transmitter, and a receiver. The transmitter sends out an IR signal; meanwhile, the receiver is waiting for that IR signal to be returned.

“Speed x Time = Distance”

The sensor records the time it takes for the signal to return and uses the formula above to calculate the distance of the object. This value is then converted to an electrical current and sent to the Arduino.

The next item of hardware that is included in our TruParking device is an Arduino Uno. The general use of this microcontroller is to run a code continuously in a loop that has been properly compiled and uploaded to the Arduino. The Arduino is a low-power consuming

microcontroller board, that is equipped with 14 digital input/output pins and 6 analog input/output pins [12].

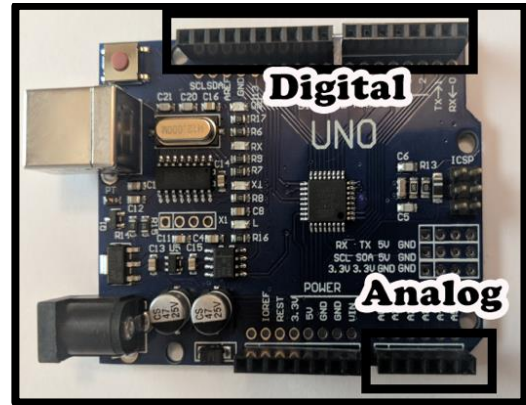


Figure 5. Arduino Uno with Digital and Analog pins labeled

Figure 5 shows the pins on the Arduino Uno. The primary reason we used this microcontroller was for the analog availability. Digital input/output pins only return a single true or false value back to the host device of the pin; this is because they only can determine a high or low in the electrical current. Analog input/output pins interpret the actual value of the electrical current the pin receives from the sensor and will return precise data to the host device. With the analog pins, we are able to receive the precise distance values from the IR sensors that our algorithm requires.

The last component that is the core of the TruParking model is the Raspberry Pi. The Raspberry Pi is a low-cost single-board microcomputer that is highly efficient for specific tasks. The Pi was built in the United Kingdom for individuals to learn basic programming and problem solving, however, it finds its usage in almost every computing approach. [13].



Figure 6. Raspberry Pi - model 3 B

Figure 6 above displays the Raspberry Pi, this single-board computer has many options for connections. There is an accessible High-Definition Multimedia Interface (HDMI) cable to use the Pi with a monitor if there is no available internet connectivity. The board includes an RJ45 Ethernet connection port, along with an 802.11 Wi-Fi chip. This provides us with the ability to connect wirelessly or via hardware to the internet with our device. These capabilities are required since our data is pushed onto the Amazon Web Service (AWS) cloud over the internet.



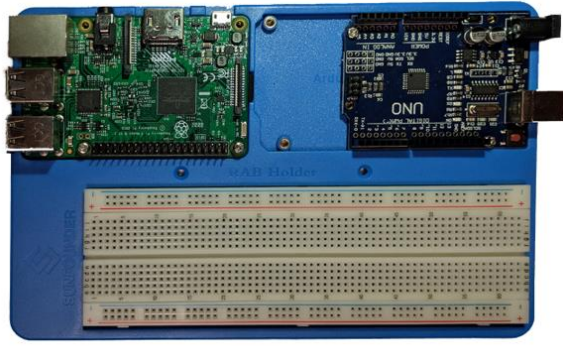


Figure 7. Top-down view of device setup

Figure 7 shows a top-down view of all the computing hardware components to build a single device. The Raspberry Pi and Arduino communicate through a single USB A to B connector wire, which also supplies adequate power to the Arduino. To connect the sensors to the Arduino we needed to use a solderless breadboard which gives access to 5 Volts of power and ground (to ensure the safety of the current) for multiple pins on both sensors. The data pins for the infrared (IR) sensors are connected to the analog pins on the Arduino. For the Pi to upload code to the Arduino the correct software needs to be installed. This software is called an Integrated Development Environment (IDE). This IDE allows the Raspberry Pi to properly upload the code to the Arduino. This gives us the ability to seamlessly compile and upload code to the Arduino to run our sensors. This code will then run endlessly on the Arduino until a new code is uploaded or the device is powered down.

The Algorithm that we uploaded to the Arduino device is represented by the pseudocode below.

```

Get Distance_A
Get Distance_B

If ( 1 < Distance_A < 450 )
    Then ( Time_A_Broke = Now )
Else
    Time_A_Unbroken = Now

If ( 1 < Distance_B < 450 )
    Then ( Time_B_Broke = Now )
Else
    Time_B_Unbroken = Now

If ( Time_A_Broke < Time_B_Broke < Time_A_Unbroken )
    Then( output: Entrance
           Spots = Spots - 1 )

If ( Time_B_Broke < Time_A_Broke < Time_B_Unbroken )
    Then( output: Exit
           Spots = Spots + 1 )

```

Figure 8. TruParking algorithm to determine whether an entrance or exit has occurred

Once the algorithm in figure 8 is running, the sensors will be able to distinguish between an entrance or exit by using our algorithm. The sensors will be configured to have two settings, broken and unbroken based on their distance reading. When the sensor is crossed/broken it will be marked with a timestamp and compared to the time of the second sensor if it has been crossed as well. If we have two sensors that are broken at the same time we know a car has traveled in front of our device and we can count this event as an entrance or exit depending on which sensor was triggered first. Once data is collected we can redirect the

output to create a file on our Raspberry Pi by using the Python GrabSerial library [14]. This file will determine whether the device has detected an entrance or exit. Every time the file is updated on the Pi it is automatically uploaded to our AWS cloud.

#### D. Amazon (AWS) Cloud

AWS is a secure cloud services platform, offering computing power, database storage, content delivery, and other functionality to help businesses scale and grow [15]. This service has given us the ability to store and analyze all the data collected with the TruParking device.

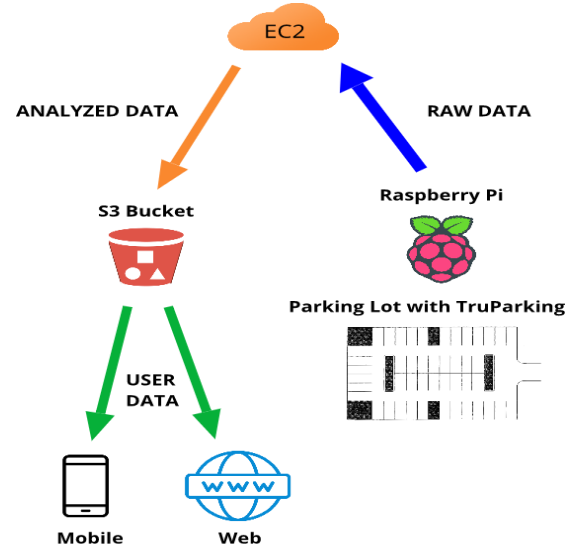


Figure 9. Diagram of TruParking's data-flow

There are many things AWS offers, for the scope of this project. We used an Amazon EC2 Ubuntu Instance and an Amazon S3 storage bucket.

An instance is a virtual computing environment that can be accessed remotely by users using an SSH command in the command terminal. For security purposes, each instance has a unique key that the user must possess to log onto the given instance.

A storage bucket is a straightforward way to store data on the web. This is the most efficient way to be able to display our data for our users. S3 is used for many mobile and web applications to access data.

##### 1) Data Storage

Figure 9 shows the Raspberry Pi is transferring all the data files to the cloud. To ensure the security of our data we used the Secure Copy (SCP) method. This method only allows devices with a specific key to log into the EC2 instance. This will confirm that our data is unable to be tampered with and that we are receiving accurate values. We then used an S3 bucket to store objects for our mobile and web interfaces to access. These objects are created by our EC2 instance after the calculations are complete.

##### 2) Analytics

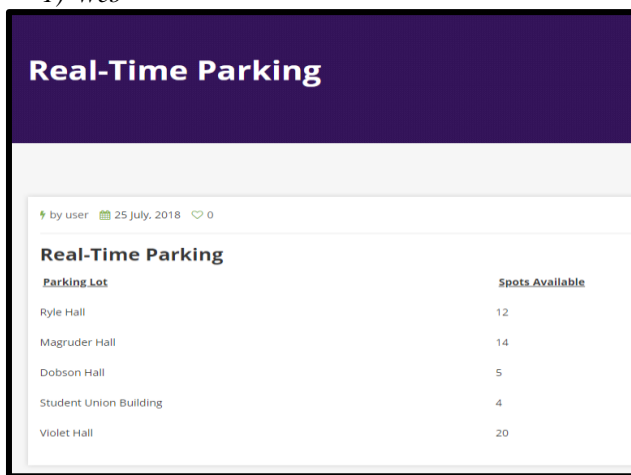
We used our EC2 Ubuntu instance for two tasks—storage and analytics. The main functionality for this virtual desktop environment is to run a Java code to analyze data files that are received from our Raspberry Pi. To produce our real-time analytics, once the code reads through the data file it will create a final text file with the number of spots

available at that current time. After that file is created it will be sent immediately to our S3 bucket. For analytics over time, we will use the timestamp that is included with every file along with the folders that are labeled with the date. Using this info, we can find the statistics for the availability in a parking lot at certain times on any given day. Likewise, all this data will be kept in our S3 bucket along with our real-time data and will then be displayed on our mobile and web user interfaces.

### E. Interfaces

In this section we will describe our two user interfaces, these were made in two ways to ensure accessibility for all individuals. We have a mobile application with the ability to be used on Android and Apple devices. The other form is our website where users can find the parking lot information they need through an internet browser.

#### 1) Web



Real-Time Parking	
Parking Lot	Spots Available
Ryle Hall	12
Magruder Hall	14
Dobson Hall	5
Student Union Building	4
Violet Hall	20

Figure 10. Screenshot of the TruParking web-interface

The website interface is more generalized for users. The main page displays a map that defaults to the location of the user's device. This displays real-time data and includes a search bar for users to find other locations they may desire. Additionally, the website will include tabs at the top to redirect to users to different pages. One page includes the data analytics where a location can be searched, and parking lots will be listed for users to choose from. Once a parking lot is chosen the user can view all data collected for the specified lot. Lastly, the website will also include a tab for information about TruParking and contact information.

#### 2) Mobile

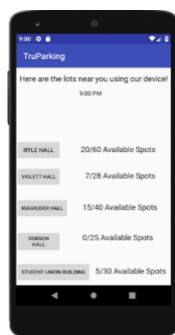


Figure 11. Screenshot of mobile application on an android device

The mobile interface for TruParking is optimized to use the user's location. With the user's location, the application can pinpoint the parking lots in range that are utilizing the TruParking device. Therefore, the user can see which parking lots have available spots or are full. This application is automatically updated to give the user the capability to view this data in real-time; along with the ability to search future times with our data analytics.

### Use Cases

One use case for our mobile application is student usage at a university. The main interest of these students would be checking for real-time spot availability to avoid parking conflicts before a class or meeting. TruParking will prevent students from being late or rushing to their destination because of parking. Eliminating these parking conflicts will create a safer driving environment for students and faculty on the roads near campus.

Another use case for our application is when an individual uses TruParking to find parking at an airport. With our data analytics, they would be able to use the TruParking app to plan for parking in advance. This planning ahead would prevent missed flights and users endlessly searching for parking inside a parking garage.

A third use case would be individuals traveling to shopping malls. The real-time analytics give the users the ability to determine the optimal time to visit stores and avoid overcrowding. Additionally, business owners would use the analytics to recognize shopping trends over time and strategize accordingly.

## IV. EVALUATION

Our method for testing our device included setting our device up in a faculty parking lot on Truman State University's campus. The experiment provided us with information on the accuracy of our device and allowed us to address issues with the exact configuration of our device. We needed to ensure that our algorithm was correctly responding to different objects passing through our break-beams and ensure that the arms of our device were covering the correct length of the entrances and exits.

TruParking requires that it sit two feet off the ground so that the infrared sensors are pointing near the mid-section of passing vehicles. To maintain the portability of our prototype we sat the main hardware on top of two cinder blocks. Then we powered our device and monitored the readings. The device initially had some errors due to conflicts in the computing speed of the three components of the device (IR sensors, Arduino Uno, and Raspberry Pi). To account for this error, we add a pause into the code to pause the polling of a sensor once a "break" has been recorded. Through extensive trial-and-error, we determined an appropriate pause interval, but we believe this to be a varying interval, as it is derived from the time it would take the average car to pass through the beams. In some lots, the entry requires the driver to turn, while others do not require turning. Differences, such as the turning or not turning into the entrance, will impact the time interval needed to configure our device properly.

In addition, we determined that the separation between our sensors was an appropriate distance. The separation is initially required to detect objects of a certain length; we ignore any objects not long enough to break both sensors

simultaneously. The distance of our sensors was chosen because it is at a length that is between the average maximum, and the minimum length of a car.

To validate the accuracy of TruParking we did validation testing over a period of a few weeks. Using the algorithm mentioned earlier in figure 8, the device prints out whether an entrance or an exit event has occurred. We refer to the output printed by TruParking as the actual data, and the data we recorded by hand the expected data. After the data collection was completed we analyzed the actual data vs. expected data, for each day. Over the period of a few weeks, the percentage error was calculated to be just under 3%. In our observations, this error occurred because of misfiring or false positive given by the IR sensors we used. This error rate, however low, could be addressed by using a more accurate IR sensor or an image analyzer in the Raspberry-Pi. This image analyzer would take pictures every time a sensor is triggered further verifying the entrance or exit event. This perhaps should be the next step for this work.

```

For every 15 minute interval
{
  If( Time_x < Time < Time_y)
  Then (
    Increment Spots_Available by 1
    Increment Count by 1)
  }
For every 15 minute interval
{
  Average = Spots_Available/Count
}

```

Figure 12. Algorithm used for TruParking analytics

Figure 12 presents the algorithm we used for analytics that provides the predictions for parking lot availability for any 15-minute interval of the day. Based on our observations we realized anything less than 15 minutes would be too fine, and anything more would be too coarse. The first line iterates through each of the 96, 15-minute intervals of the day. For each of the intervals, we check to see if the data in our data file corresponds to the current time interval in question; if so, we add the related "Spots\_Available" data to a running sum for the interval, and we increment the running count for the interval by 1. To calculate the average spots available for the 15-minute interval, we divide the running sum by the count of data entries. The average will then be plotted on a graph showing the average number of spots available at each 15-minute interval.

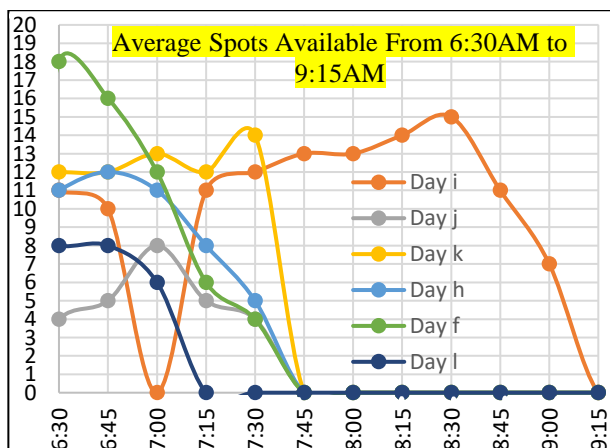


Figure 13. Graph of TruParking Analytics

Figure 13 displays the TruParking data analytics collected over time at a University parking lot. This data was analyzed and used to determine the average number of spots available during certain times of the morning. Our users will have this information as a prediction of the number of spots available at any given time on any given day.

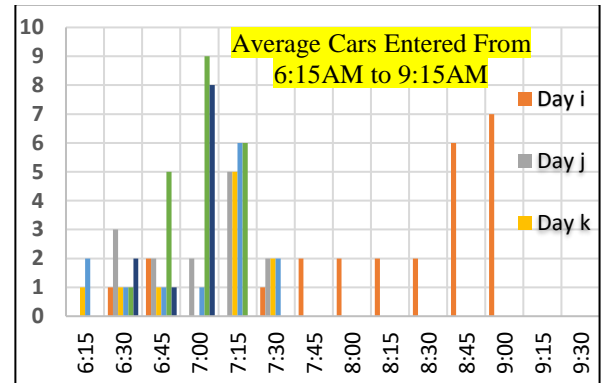


Figure 14. Graph of TruParking Analytics

Figure 14 displays another service of TruParking analytics. In addition to providing information about the number of available spots, TruParking analytics can provide data about the number of cars to enter a parking lot at a given time. This provides the user with parking lot traffic information. This information would be valuable to the lot owner as well, giving them information about its usage. Drivers will still find this information useful because it gives them information about how many cars are usually entering the parking lot at a certain time, so they will know how many cars are likely heading to this lot at a given time.

## V. FUTURE WORK

In this section, we will discuss future applications for the TruParking device. These additions could increase the value and versatility of the device.

The work in this paper contributes to solving parking issues that are commonplace in today's world. While this work does a great amount to address these issues, there is room for improvement. Additional functionalities that could be added to this work include, QR or RFID scanning, image processing, spot reservation, etc. These methods would make the data even more precise for TruParking. This device has a lot of potential, and in the future, this could be used in other operations in addition to finding parking.

This device could potentially have use in crime solving such as car break-ins or stolen vehicles. To track these vehicles, we could connect camera sensors to the Pi along with image processing software. This would equip us with the ability to, as vehicles enter or exit, take pictures of license plate numbers and/or drivers. Each vehicle would have a photo with a unique timestamp which would validate the amount of time that the car was in the parking lot. This could be useful for law enforcement in many ways. Arguing over parking tickets and if a non-registered vehicle parked in a reserved spot would be a thing of the past. This could save everyone time and resources, and that is what the TruParking device is all about. We see many

avenues to make TruParking a top-tier smart parking device for the present day and beyond.

## VI. CONCLUSION

In conclusion, our work discusses the long-term wastefulness and economic strain that comes from searching for parking spaces. We presented data supporting our claim that parking has been and is going to continue to be a problem for the foreseeable future. The data that we provided also suggests that if trends continue, the problem will not only remain but increase in severity. Our work addresses these parking issues and presents a device that is simple to use, simple to install, and maintains an affordable design. Once installed, users will access TruParking via our mobile application or a web interface. TruParking will provide real-time data and analytics to the user in a straightforward, easily accessible manner.

## VII. ACKNOWLEDGMENT

This work is partially supported by the Computer Science department and the TruScholars research program at Truman State University.

## VIII. REFERENCES

- [1] Federal Highway Administration, "Highway Finance Data Collection," U.S. Department of Transportation, 7 November 2014. [Online]. Available: <https://www.fhwa.dot.gov/policyinformation/pubs/hf/pl11028/cchapter4.cfm>. [Accessed 8 August 2018].
- [2] Federal Highway Administration, "State Motor-Vehicle Registrations - 2016," 2017 November. [Online]. Available: <https://www.fhwa.dot.gov/policyinformation/statistics/2016/mv1.cfm>. [Accessed 8 August 2018].
- [3] Statista, "U.S. automobile registrations from 2000 to 2016," 2017. [Online]. Available: <https://www.statista.com/statistics/192998/registered-passenger-cars-in-the-united-states-since-1975/>. [Accessed 8 August 2018].
- [4] K. McCoy, "Drivers spend an average of 17 hours a year searching for parking spots," 12 July 2017. [Online]. Available: <https://www.usatoday.com/story/money/2017/07/12/parking-pain-causes-financial-and-personal-strain/467637001/>. [Accessed 8 August 2018].
- [5] INRIX, "Searching for Parking Costs Americans \$73 Billion a Year," [Online]. Available: <http://inrix.com/press-releases/parking-pain-us/>. [Accessed 8 August 2018].
- [6] Indect, "What We Do," Indect, [Online]. Available: <http://indect.com/what-we-do/>. [Accessed 17 August 2018].
- [7] M. Winter and J. Osterwiel, "Apparatus and method for sensing the occupancy status of parking spaces in a parking lot". United States of America Patent 7,116,246, 3 October 2006.
- [8] T. N. PHAM and D. B. N. D.-J. D. MING-FONG TSAI, "A Cloud-Based Smart-Parking System Based on Internet-of-Things Technologies," IEEE Access, vol. 3, pp. 1581-1591, 2015.
- [9] A. Khanna and R. Anand, "IoT based Smart Parking System," IEEE, pp. 266-270, 2016.
- [10] J. Geerling, "Raspberry Pi Dramble," [Online]. Available: <https://www.pidramble.com/wiki/benchmarks/power-consumption>. [Accessed 19 August 2018].
- [11] "ArduinoCC," [Online]. Available: <https://store.arduino.cc/arduino-uno-rev3>. [Accessed 18 August 2018].
- [12] "Arduino Uno Rev3," Arduino, [Online]. Available: <https://store.arduino.cc/usa/arduino-uno-rev3>. [Accessed 15 August 2018].
- [13] Raspberry Pi Foundation, "About Us," [Online]. Available: <https://www.raspberrypi.org/about/>. [Accessed 17 August 2018].
- [14] "Grabserial," 13 January 2017. [Online]. Available: <https://elinux.org/Grabserial>. [Accessed 12 August 2018].
- [15] Amazon, "What Is AWS," Amazon, [Online]. Available: <https://aws.amazon.com/what-is-aws/>. [Accessed 18 August 2018].
- [16] "AdaFruit," [Online]. Available: <https://www.adafruit.com/product/1568>. [Accessed 8 August 2018].